Debugging

Announcements

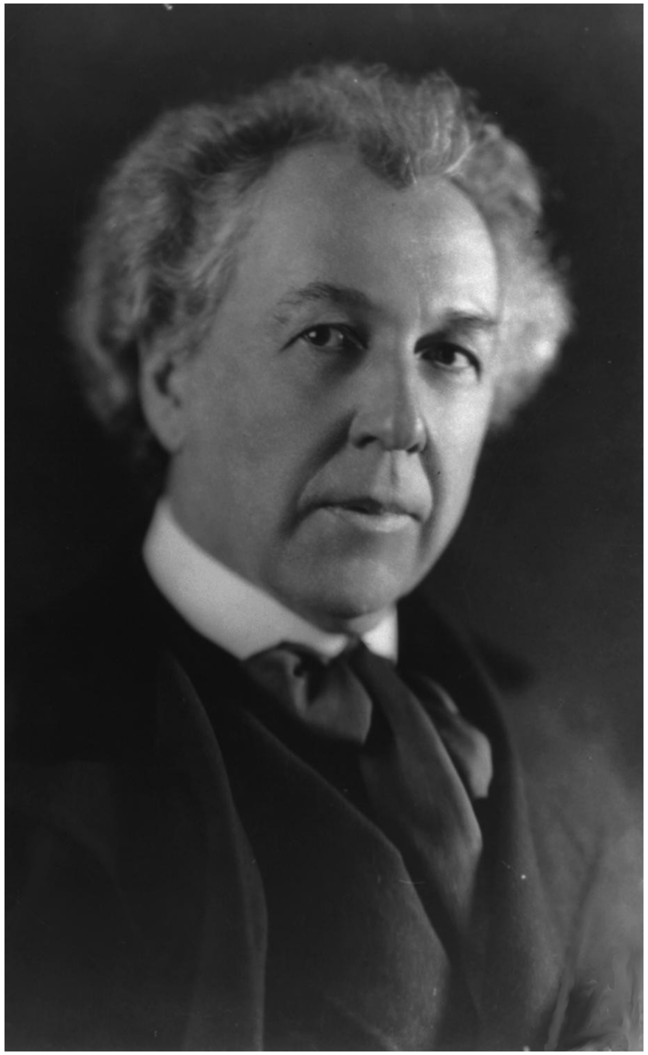
- Assignment 4 due right now.
- Assignment 5 (Yahtzee!) out, due next **Wednesday, February 29**.
  - Cool way to play around with arrays and algorithms.
  - Hone your skills with data processing.
  - YEAH hours **Thursday, 7-8PM** in Braun Lecture Hall.
- Midterm graded; will be returned at the end of lecture.
  - Details at end of lecture.
  - Midterms will be outside of Gates 178 after they've been returned in lecture.

Yahtzee Demo

Debugging

http://en.wikipedia.org/wiki/File:First_flight2.jpg

Designing the Program



- Think like an **architect**.
- What is the grand vision?
- What will the large pieces be?

http://upload.wikimedia.org/wikipedia/commons/9/99/Frank_Lloyd_Wright_LC-USZ62-36384.jpg

Writing the Program

- Think like an **engineer**.
- Flesh out the design by actually making it happen.

http://www.infobarrel.com/media/image/15873.jpg

Testing the Program

- Think like a **vandal**.
- Try doing things to the program that aren't expected:
  - Enter invalid or nonsensical data.
  - Don't follow directions.

http://www.conversationmarketing.com/Snidely%2BWhiplash.png

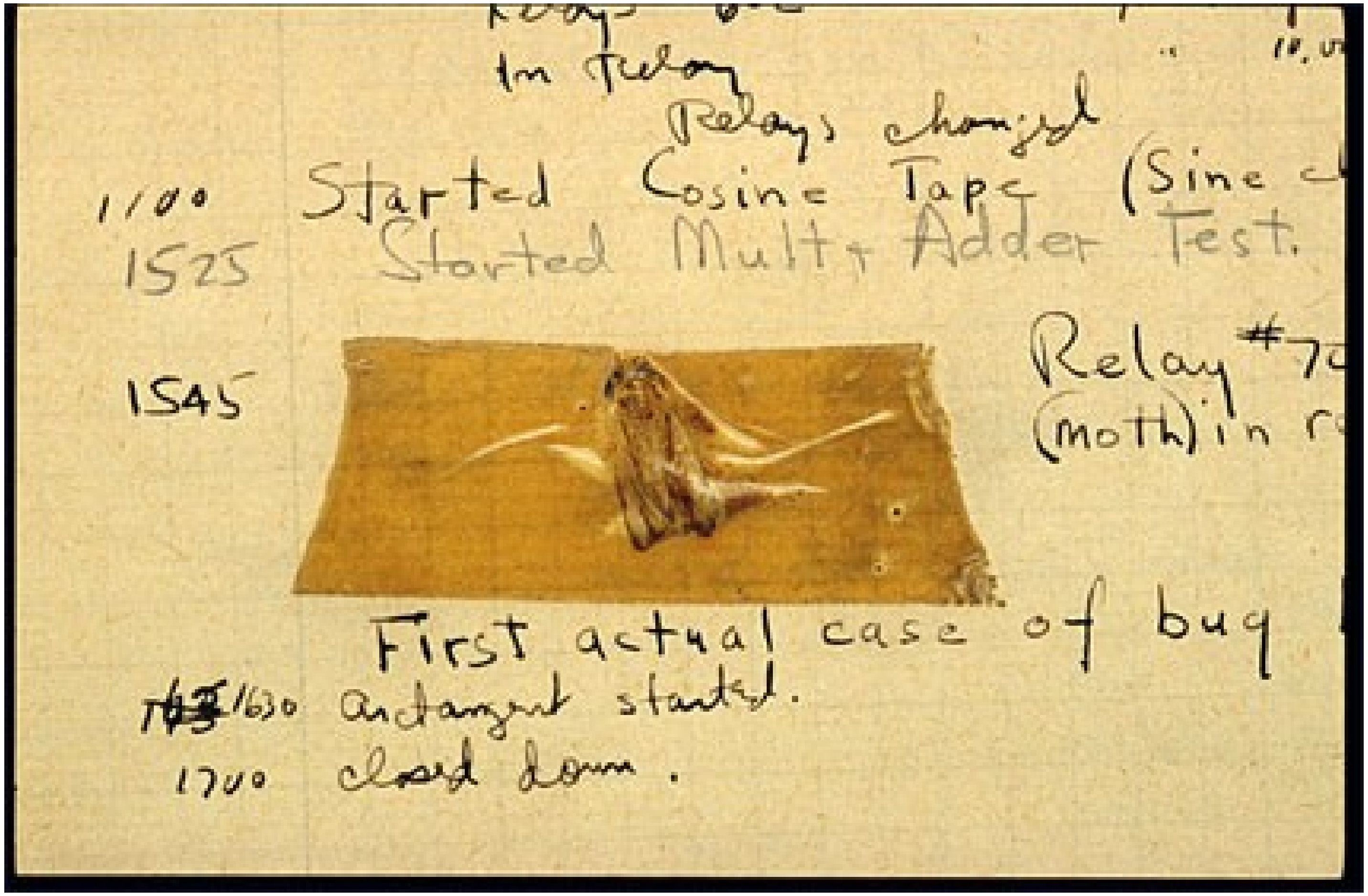
Debugging the Program

- Think like a **detective**.
- Follow the clues the program gives to determine the logical cause of the bug.

http://upload.wikimedia.org/wikipedia/commons/c/cf/Agatha_Christie.png

What Causes Bugs?

Actual Bug from Mark II Computer



in relay

Relays changed

1100 Started Cosine Tape (Sine

1525 Started Mult + Adder Test.

1545 Relay #7

(moth) in r

First actual case of bug

~~143~~ 1630 Antangent started.

1700 closed down.

© National Museum of American History

Source: Smithsonian National Museum of History
Kenneth E. Behring Center

`http://americanhistory.si.edu/dynamic/images/collections_xlarge/92-13129_428px.jpg`

What Causes Bugs?

- Incorrect values in variables.
  - Using the wrong variable.
  - Computing a value incorrectly.
- Logical errors.
  - Looping the wrong number of times.
  - Incorrect expressions in `if` statements.
- Bad assumptions.
  - Assuming that the input has some form that it doesn't.

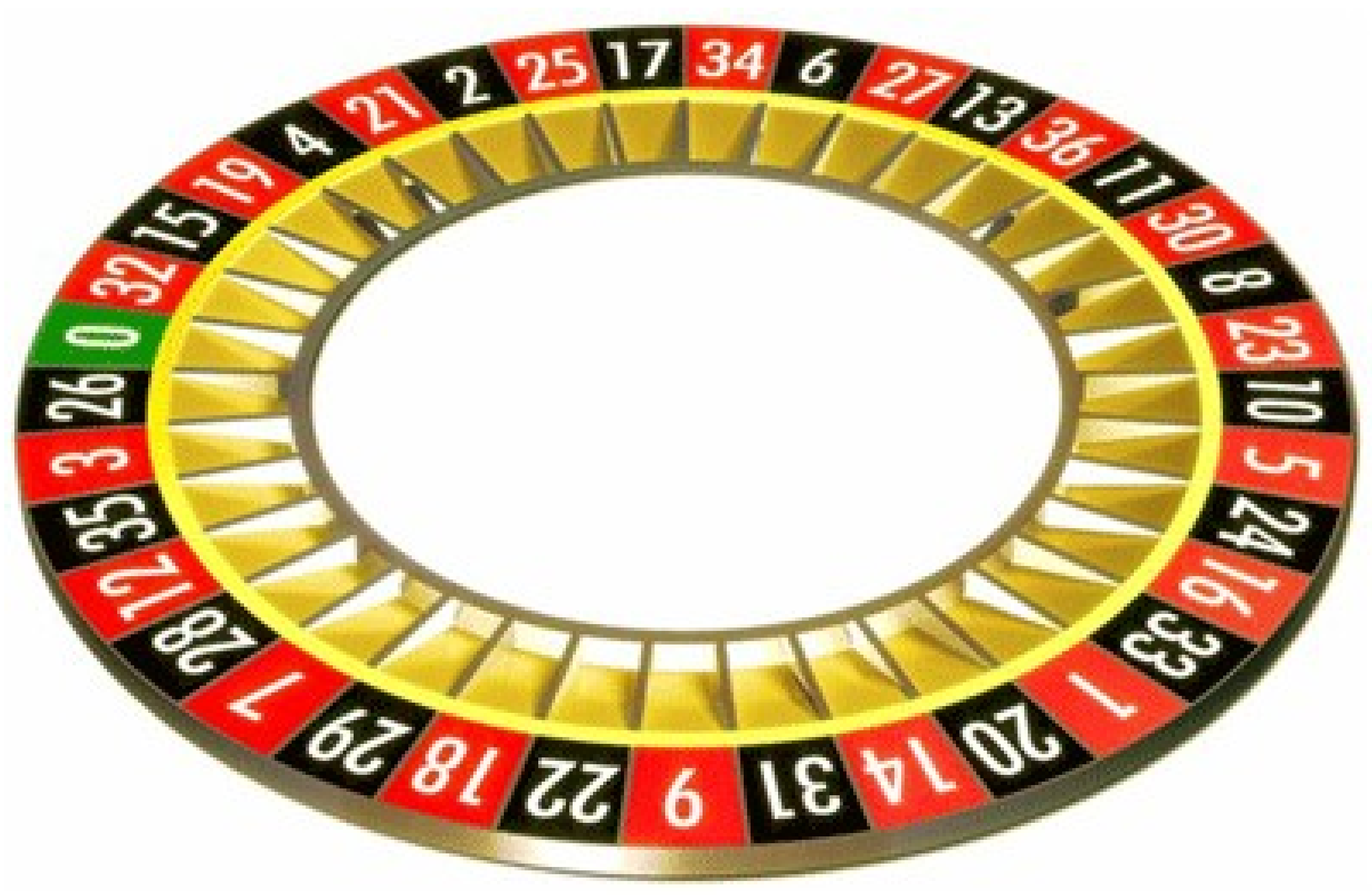
Debugging Philosophy

- Find out what the program **is** doing, not what it's **not** doing.
  - The computer will do exactly what you told it to do; you just told it to do the wrong thing!
- Be patient: The bug isn't trying to hide, and with enough effort you're going to find it.

While Debugging...

- Don't start making changes to the program without a good reason.
  - You're going to introduce new bugs!
  - You're going to complicate your bug hunt!
- Ask the program to tell you what it's doing.
  - Pull up a **debugger** and look at what's happening.

An Example



http://www.casinosonline.co.uk/userContent/roulette-pic.gif

Roulette

- Wheel contains the numbers 0 – 36.
- A ball is tossed into the wheel and ends at one of the numbers.
- Can place lots of different bets on the outcome, but we'll consider four:
  - **Low**: The number is between 1 and 19.
  - **High**: The number is between 20 and 36.
  - **Odd**: The number is odd.
  - **Even**: The number is even (but not zero).
- If you win, you get 2x your bet back.
  - Odds are slightly against you because 0 always loses.

Runtime Exceptions

Runtime Exceptions

- **`ArrayIndexOutOfBoundsException`**
  - Attempted to look up an element in an array at an invalid index.
  - Check to make sure that the index is valid and that the array has the length you think it does.

Runtime Exceptions

- **`ArrayIndexOutOfBoundsException`**
  - Attempted to look up an element in an array at an invalid index.
  - Check to make sure that the index is valid and that the array has the length you think it does.
- **`StringIndexOutOfBoundsException`**
  - Attempted to look up an element in a **`String`** at an invalid index.
  - Check to make sure that the index is valid and that the **`String`** has the length you think it does.

Runtime Exceptions

- **`ArrayIndexOutOfBoundsException`**
  - Attempted to look up an element in an array at an invalid index.
  - Check to make sure that the index is valid and that the array has the length you think it does.
- **`StringIndexOutOfBoundsException`**
  - Attempted to look up an element in a **`String`** at an invalid index.
  - Check to make sure that the index is valid and that the **`String`** has the length you think it does.
- **`NullPointerException`**
  - Attempted to call a method on a **`null`** reference (for example, an uninitialized **`String`** or **`GRect`**).
  - Check the receiver object to make sure it's not **`null`**.

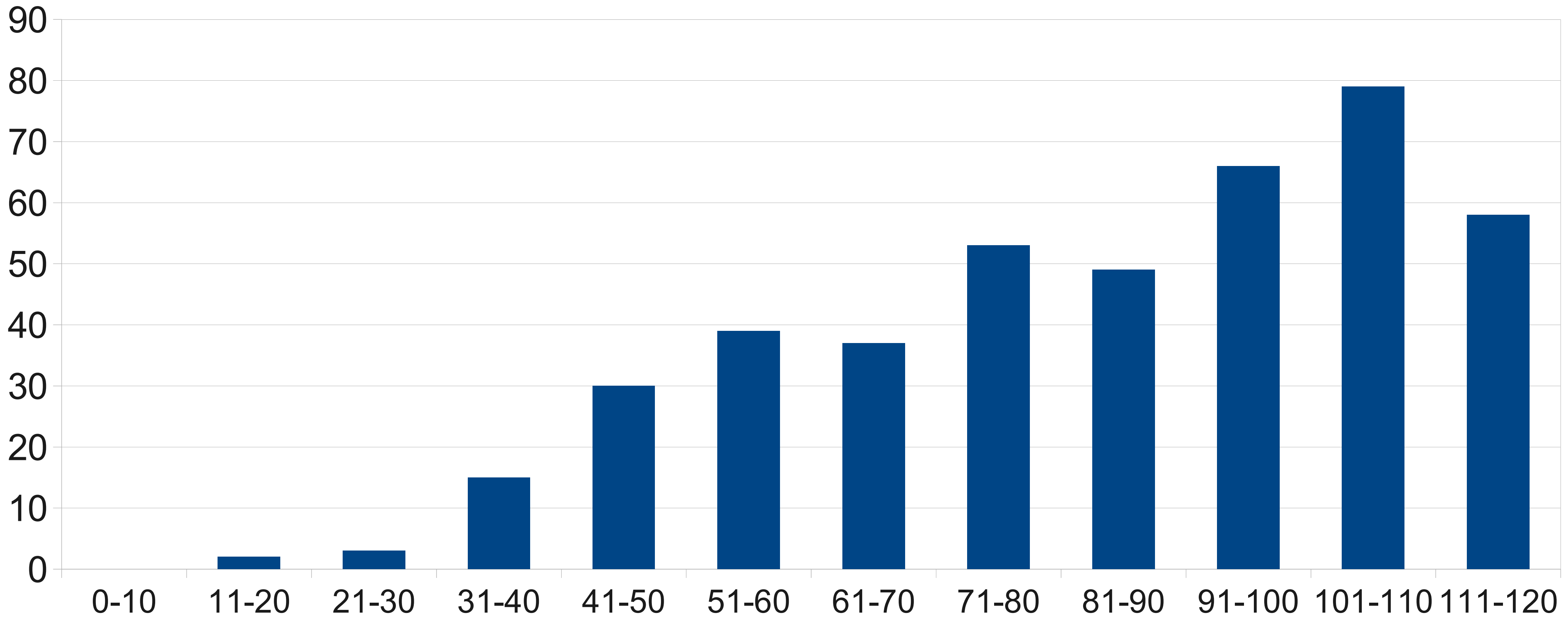
Infinite Loops

- Infinite loops result when a loop that ought to terminate never does.
- Program will seem unresponsive, or will keep doing the same thing over and over again.
- Step through the program with a debugger.
  - Can you find out why the loop isn't terminating?

Preventing Bugs

- The best way to debug is to prevent bugs from occurring in the first place.
- **Test your program often**.
    - Write the program in small pieces and verify that each piece works as you write it.
    - Sometimes called “unit testing.”
- **Use libraries when possible**.
    - Thoroughly-tested code is less likely to be buggy than your own version.

Midterm Scores



Mean:	**83.7/120**	**(70%)**
Median:	**88/120**	**(73%)**
Stdev:	**24**	